Table of Content

Overview

Inventory Management System is a platform that aims to track products in user's stack and get updated about the orders they receive. It provides a comprehensive solution to managing products, suppliers, and orders. This Inventory Management System enables users to track product availability, manage suppliers, and process orders.

This product is centered towards an Entity who manages multiple products from various suppliers. A user can create orders for the customers which results in deducting quantities from the product table. At the same time, the product restock feature allows the owner to restock products as the stock depletes. This ensures that the stock never empties out at the owner's end.

This project will focus on developing a backend for inventory management system with RESTful APIs, utilizing Spring Boot and Hibernate for database interactions. The system will not include a user interface; instead, it will be tested and validated using Postman to ensure API functionality and reliability.

The core functionalities includes:

- User Management
- CRUD operations for products, suppliers, and orders.
- User authentication and authorization using JWT.
- Data import and export functionalities for products and orders in CSV format.
- Implementation of various relationships to model the database schema.

For more such as Database design and working of jwt based authentication, click here.

Base URL

The base URL for all API requests is: `http://localhost:8080/api`

Authentication

Login Endpoint

POST /auth/login

Validates user credentials and returns basic user details and a jwt token.

Response

Returns a JSON object with the following properties:

- `message`: token
- `data`: A JSON object of user details
  - `userId`: The Id of the user.
  - `fullname`: The full name of the user.
  - `email`: The email of the user.
  - `roles`: An array containing roles assigned to the user.
  - `jwtToken`: The jwt token generated for this login.

Example

Request:

```
POST /auth/login
```

Body:

```
{
"email": "ak@gmail.com",
"password": "asdf"
}
```

Response: `200 OK`

```
{
    "message": "token",
    "data": {
        "userId": 1,
        "fullname": "Aashish Katwal",
        "email": "ak@gmail.com",
        "roles": [
            "EMPLOYEE",
            "ADMIN"
        ],
        "jwtToken":
"eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJha0BnbWFpbC5jb20iLCJp
YXQiOjE3MzIyNTc2NjQsImV4cCI6MTczMjI2NDg2NH0.-zIBDKiM5aScimSf9J1VE-JLgC
TwT_3daZJ4CnlL_RQ"
    }
}
```

Errors

This API uses the following error codes:

- `401 Unauthorized`: The credentials are invalid.
- `500 Internal Server Error`: An unexpected error occurred on the server.

Signup Endpoint

POST /auth/signup

Registers user details and returns basic user details.

Response

Returns a JSON object with the following properties:

- `message`: User created successfully.
- `data`: A JSON object of user details
    - `userId`: The Id of the user.
    - `fullname`: The full name of the user.
    - `email`: The email of the user.
    - `roles`: An array containing roles assigned to the user.
    - `jwtToken`: null

Example

Request:

```
POST /auth/signup
```

Body:

```
{
      "fullname": "Aashish Katwal",
      "email": "ak@gmail.com",
      "password": "asdf",
      "roles": ["EMPLOYEE"]
}
```

Response: `201 Created`

```
{
    "message": "token",
    "data": {
        "userId": 1,
        "fullname": "Aashish Katwal",
        "email": "ak@gmail.com",
        "roles": [
            "EMPLOYEE",
        ],
        "jwtToken": null
    }
}
```

Errors

This API uses the following error codes:

- `400 Bad Request:`Duplicate Record. Email already exists.
- `401 Unauthorized`: Missing authentication token.
- `500 Internal Server Error`: An unexpected error occurred on the server.

Product

New Product Endpoint

POST /secure/product/new

Saves new product to the database.

Response

Returns a JSON object with the following properties:

- `message`: Product Added Successfully.
- `data`: A JSON object of product details
  - `productId`: The Id of the product.
  - `productName`: The name of the product.
  - `description`: The description of the product.
  - `price`: The price of the product.
  - `quantity`: The quantity of available product.
  - `supplierName`: The name of the supplier.

Example

Request:

```
POST /secure/product/new
```

Body:

```
{
      "productName": "Asus CPU Casing",
      "description": "Metal Panel with RGB lights",
      "price": 60000,
      "quantity": 2,
      "supplier": {“supplierId": 2}
}
```

Response: `201 Created`

```
{
    "message": "Product Added Successfully",
    "data": {
        "productId": 12,
        "productName": "Asus CPU Casing",
        "description": "Metal Panel with RGB lights",
        "price": 60000,
        "quantity": 2,
        "supplierName": "Acer Inc."
    }
}
```

Errors

This API uses the following error codes:

- `400 Bad Request:` Resource not found for missing invalid values such as negative value of quantity or missing field.
- `401 Unauthorized`: Missing authentication token.
- `500 Internal Server Error`: An unexpected error occurred on the server.

Get Products Endpoint

GET /secure/product/all

Fetches a list of products from the database.

Response

Returns a JSON object with the following properties:

- `message`: Products fetched successfully.
- `data`: A JSON object of product details
    - `productId`: The Id of the product.
    - `productName`: The name of the product.
    - `description`: The description of the product.
    - `price`: The price of the product.
    - `quantity`: The quantity of available product.
    - `supplierName`: The name of the supplier.

Example

Request:

`GET /secure/product/all`

Body: N/A

Response: `200 OK`

```
{
    "message": "Fetched products successfully",
    "data": [
        {
            "productId": 1,
            "productName": "Acer Nitro 2024",
            "description": "High-performance gaming laptop with 16GB RAM",
            "price": 1200.50,
            "quantity": 110,
            "supplierName": "Asus INFOTek"
        },
        {
            "productId": 2,
            "productName": "Asus Zenbook 2020",
            "description": "High-performance laptop for Designers",
            "price": 120000.00,
            "quantity": 30,
            "supplierName": "Acer Inc."
        }
    ]
}
```

Errors

This API uses the following error codes:

- `401 Unauthorized`: Missing authentication token.
- `500 Internal Server Error`: An unexpected error occurred on the server.

Product By Id Endpoint

GET /secure/product/{productId}

Fetches product for given ID from the database.

Response

Returns a JSON object with the following properties:

- `message`: Products fetched successfully.
- `data`: A JSON object of product details
    - `productId`: The Id of the product.
    - `productName`: The name of the product.
    - `description`: The description of the product.
    - `price`: The price of the product.
    - `quantity`: The quantity of available product.
    - `supplierName`: The name of the supplier.

Example

Request:

```
GET /secure/product/2
```

Body: N/A

Response: `200 OK`

```
{
    "message": "Fetched products successfully",
    "data": {
        "productId": 2,
        "productName": "Asus Zenbook 2020",
        "description": "High-performance laptop for Designers",
        "price": 120000.00,
        "quantity": 30,
        "supplierName": "Acer Inc."
    }
}
```

Errors

This API uses the following error codes:

- `400 Bad Request`: Resource not found for invalid product Id.
- `401 Unauthorized`: Missing authentication token.
- `500 Internal Server Error`: An unexpected error occurred on the server.

Delete Product Endpoint

DELETE /secure/product/delete/{productId}

Deletes a product by id.

Response

Returns a JSON object with the following properties:

- `message`: Product Added Successfully.
- `data`: A JSON object of project details
  - `productId`: The Id of the product.
  - `productName`: The name of the product.
  - `description`: The description of the product.
  - `price`: The price of the product.
  - `quantity`: The quantity of available product.
  - `supplierName`: The name of the supplier.

Example

Request:

```
DELETE /secure/product/delete/2
```

Body: N/A

Response: `204 No Content`

Errors

This API uses the following error codes:

- `400 Bad Request`: Resource not found for invalid product Id.
- `401 Unauthorized`: Missing authentication token.
- `500 Internal Server Error`: An unexpected error occurred on the server.

Update Product Endpoint

POST /secure/product/update/{productId}

Updates a product by id.

Response

Returns a JSON object with the following properties:

- `message`: Product Added Successfully.
- `data`: A JSON object of project details
    - `productId`: The Id of the product.
    - `productName`: The name of the product.
    - `description`: The description of the product.
    - `price`: The price of the product.
    - `quantity`: The quantity of available product.
    - `supplierName`: The name of the supplier.

Example

Request:

```
POST /secure/product/update/1
```

Body:

```
{
    "productName": "Asus Zenbook 2021",
    "description": "High-performance laptop for Designers",
    "price": 120000.00,
    "quantity": 13,
    "supplier": {
        "supplierId": 2
    }
}
```

Response: `200 OK`

```
{
    "message": "Product fetched successfully",
    "data": {
        "productId": 2,
        "productName": "Asus Zenbook 2021",
        "description": "High-performance laptop for Designers",
        "price": 120000.00,
        "quantity": 13,
        "supplierName": "Acer Inc."
    }
}
```

Errors

This API uses the following error codes:

- `400 Bad Request`: Resource not found for invalid product Id.
- `401 Unauthorized`: Missing authentication token.
- `500 Internal Server Error`: An unexpected error occurred on the server.

Restock Product Endpoint

POST /secure/product/restock

Imitates product restock. This increases product quantity in the database.

Response

Returns a JSON object with the following properties:

- `message`: Product Added Successfully.
- `data`: A JSON object of project details
    - `productId`: The Id of the product.
    - `productName`: The name of the product.
    - `description`: The description of the product.
    - `price`: The price of the product.
    - `quantity`: The new quantity of the product.
    - `supplierName`: The name of the supplier.

Example

Request:

```
POST /secure/product/restock/
```

Body:

```
{
    "productId": 2,
    "quantity": 40
}
```

Response: `200 OK`

```
{
    "message": "Product fetched successfully",
    "data": {
        "productId": 2,
        "productName": "Asus Zenbook 2021",
        "description": "High-performance laptop for Designers",
        "price": 120000.00,
        "quantity": 140,
        "supplierName": "Acer Inc."
    }
}
```

Errors

This API uses the following error codes:

- `400 Bad Request`: Negative quantity or invalid product Id.
- `401 Unauthorized`: Missing authentication token.
- `500 Internal Server Error`: An unexpected error occurred on the server.

CSV Product Import Endpoint

POST /secure/product/import

Imports product from the CSV

Response

Returns a JSON object with the following properties:

- `message`: Success
- `data`: Success or error message.

Request:

```
POST /secure/product/import/
```

Body form-data:

```
key: productCsv
Value: productData.csv
```

Response: `200 OK`

```
{
    "message": "Success",
    "data": "Successfully imported 3 products."
}
```

Errors

This API uses the following error codes:

- `400 Bad Request`: Negative quantity or missing supplier.
- `401 Unauthorized`: Missing authentication token.
- `500 Internal Server Error`: An unexpected error occurred on the server.

CSV Product Export Endpoint

GET /secure/product/export

Exports the list of products available in the database.

Response

Downloads the products as CSV and saves to the user's system.

Example

Request:

```
GET /secure/product/export
```

Body: N/A

Response: `200 OK`

```
"As of 2024-11-22 01:27:32 PM: "
"ID","Product Name","Description","Price","Quantity","Supplier"
"1","Acer Nitro 2024","High-performance gaming laptop with 16GB
RAM","1200.50","115","Asus INFOTek"
"2","Asus Zenbook 2021","High-performance laptop for
Designers","120000.00","13","Acer Inc."
"3","Asus Vivobook 2024","Medium Segment laptop for general
use","90000.00","40","Asus INFOTek"
```

Errors

This API uses the following error codes:

- `401 Unauthorized`: Missing authentication token.
- `500 Internal Server Error`: An unexpected error occurred on the server.

Supplier

New Supplier Endpoint

POST /secure/supplier/new

Saves new suppliers to the database.

Response

Returns a JSON object with the following properties:

- `message`: Supplier Added Successfully.
- `data`: A JSON object of supplier details
  - `supplierId`: The Id of the supplier.
  - `supplierName`: The name of the supplier.
  - `email`: The email of the supplier.
  - `phoneNumber`: The contact number of the supplier.
  - `status`: The status of the supplier.

Example

Request:

```
POST /secure/supplier/new
```

Body:

```
{
    "supplierName": "HP Inc",
    "email": "support@hp.com",
    "phoneNumber": "9800123456",
    "status": "ACTIVE"
}
```

Response: `201 Created`

```
{
    "message": "Supplier Added Successfully",
    "data": {
        "supplierId": 11,
        "supplierName": "HP Inc",
        "email": "support@hp.com",
        "phoneNumber": "9800123456",
        "status": "ACTIVE"
    }
}
```

Errors

This API uses the following error codes:

- `400 Bad Request:` Missing field values.
- `401 Unauthorized`: Missing authentication token.
- `500 Internal Server Error`: An unexpected error occurred on the server.

Get All Supplier Endpoint

POST /secure/supplier/

Returns the list of all suppliers in the database.

Response

Returns a JSON object with the following properties:

- `message`: Product Added Successfully.
- `data`: An array of JSON objects of supplier details
    - `supplierId`: The Id of the supplier.
    - `supplierName`: The name of the supplier.
    - `email`: The email of the supplier.
    - `phoneNumber`: The contact number of the supplier.
    - `status`: The status of the supplier.

Example

Request:

```
GET /secure/supplier/
```

Body: N/A

Response: `200 OK`

```
{
    "message": "Suppliers fetched successfully",
    "data": [
        {
            "supplierId": 12,
            "supplierName": "",
            "email": "support@hp.com",
            "phoneNumber": "9800123456",
            "status": "ACTIVE"
        },
        {
            "supplierId": 13,
            "supplierName": "lenovo",
            "email": "support@lenovo.com",
            "phoneNumber": "9800123456",
            "status": "ACTIVE"
        }
    ]
}
```

Errors

This API uses the following error codes:

- `401 Unauthorized`: Missing authentication token.
- `500 Internal Server Error`: An unexpected error occurred on the server.

Products By Supplier Endpoint

GET /secure/supplier/products?supplierId={id}

Returns the list of products of the supplier.

Parameters

- `supplierId`: The id of an existing supplier.

Response

Returns a JSON object with the following properties:

- `message`: Supplier Added Successfully.
- `data`: A JSON object of supplier details
  - `productName`: The name of the product.
  - `price`: The price of the product.
  - `supplierName:` The name of the supplier.

Example

Request:

```
GET /secure/supplier/products?supplierId=2
```

Body: N/A

Response: `200 OK`

```
{
    "message": "Products fetched successfully",
    "data": [
        {
            "productName": "Asus Zenbook 2021",
            "price": 120000.00,
            "supplierName": "Acer Inc."
        },
        {
            "productName": "Paracetamol",
            "price": 10.50,
            "supplierName": "Acer Inc."
        }
    ]
}
```

Errors

This API uses the following error codes:

- `204 No Content:` Supplier doesn't exist.
- `401 Unauthorized`: Missing authentication token.
- `500 Internal Server Error`: An unexpected error occurred on the server.

Change Supplier Status Endpoint

GET /secure/supplier/changeStatus/{supplierId}?newStatus={value}

Changes the status of the supplier in the database.

Parameters

- `newStatus`: The new status to be assigned

Response

Returns a JSON object with the following properties:

- `message`: Supplier Added Successfully.
- `data`: A JSON object of supplier details
  - `supplierId`: The Id of the supplier.
  - `supplierName`: The name of the supplier.
  - `email`: The email of the supplier.
  - `phoneNumber`: The contact number of the supplier.
  - `status`: The status of the supplier.

Example

Request:

```
GET /secure/supplier/changeStatus/1?newStatus=ACTIVE
```

Body: N/A

Response: `200 OK`

```
{
    "message": "Supplier status changed successfully!",
    "data": {
        "supplierId": 1,
        "supplierName": "Asus INFOTek",
        "email": "contact@asus.com",
        "phoneNumber": "9800112233",
        "status": "ACTIVE"
    }
}
```

Errors

This API uses the following error codes:

- `400 Bad Request:` Supplier not found.
- `401 Unauthorized`: Missing authentication token.
- `500 Internal Server Error`: An unexpected error occurred on the server.

Update Supplier Endpoint

PUT /secure/supplier/update/{supplierId}

Updates new supplier details in the database.

Response

Returns a JSON object with the following properties:

- `message`: Supplier Added Successfully.
- `data`: A JSON object of supplier details
    - `supplierId`: The Id of the supplier.
    - `supplierName`: The name of the supplier.
    - `email`: The email of the supplier.
    - `phoneNumber`: The contact number of the supplier.
    - `status`: The status of the supplier.

Example

Request:

```
PUT /secure/supplier/update/1
```

Body:

```
{
    "supplierName": "ASUSTek Computer Inc.",
    "email": "contacts@asus.org",
    "phoneNumber": "9800112233",
    "status": "ACTIVE"
}
```

Response: `200 OK`

```
{
    "message": "Updated successfully",
    "data": {
        "supplierId": 1,
        "supplierName": "ASUSTek Computer Inc.",
        "email": "contacts@asus.org",
        "phoneNumber": "9800112233",
        "status": "ACTIVE"
    }
}
```

Errors

This API uses the following error codes:

- `400 Bad Request:` Missing field values.
- `401 Unauthorized`: Missing authentication token.
- `500 Internal Server Error`: An unexpected error occurred on the server.

CSV Supplier Import Endpoint

POST /secure/supplier/import

Imports suppliers from the CSV

Response

Returns a JSON object with the following properties:

- `message`: Success
- `data`: Success or error message.

Request:

```
POST /secure/supplier/import/
```

Body form-data:

```
key: supplierCsv
Value: supplierData.csv
```

Response: `200 OK`

```
{
    "message": "Success",
    "data": "Successfully imported 5 suppliers."
}
```

Errors

This API uses the following error codes:

- `400 Bad Request`: Negative quantity or missing supplier.
- `401 Unauthorized`: Missing authentication token.
- `500 Internal Server Error`: An unexpected error occurred on the server.

CSV Supplier Export Endpoint

GET /secure/supplier/export

Exports the list of suppliers available in the database.

Response

Downloads the list of suppliers as CSV and saves to the user's system.

Example

Request:

```
GET /secure/supplier/export
```

Body: N/A

Response: `200 OK`

```
"As of 2024-11-22 01:58:30 PM: "
"ID","Supplier Name","Email","Phone Number","Status"
"1","ASUSTek Computer Inc.","contacts@asus.com","9800112233","ACTIVE"
"2","Acer Inc.","contact@acer.com","9800123349","ACTIVE"
```

Errors

This API uses the following error codes:

- `401 Unauthorized`: Missing authentication token.
- `500 Internal Server Error`: An unexpected error occurred on the server.

Orders

New Order Endpoint

POST /secure/product/new

Creates a new order.

Response

Returns a JSON object with the following properties:

- `message`: Product Added Successfully.
- `data`: A JSON object of product details
  - `orderId`: The Id of the order.
  - `totalAmount`: The total value of the order.
  - `status`: The status of the order.
  - `products`: The array of JSON objects of products
    - `quantity`: The quantity of products in orderItem.
    - `productId`: The id of the product ordered.
    - `productName`: The name of the product ordered.
    - `rate`: The rate of each product ordered.
    - `totalPrice`: The total value of each product ordered.
      Value=quantity*rate.

Example

Request:

```
POST /secure/product/new
```

Body:

```
{
    "orderedItems": [
        {
            "productId": 9,
            "quantity": 10
        }
    ]
    ,"orderStatus": "IN_PROGRESS"
}
```

Response: `201 Created`

```
{
    "message": "Order Placed Successfully",
    "data": {
        "orderId": 4,
        "totalAmount": 105.00,
        "status": "IN_PROGRESS",
        "products": [
            {
                "quantity": 10,
                "productId": 9,
                "productName": "Paracetamol",
                "rate": 10.50,
                "totalPrice": 105.00
            }
        ]
    }
}
```

Errors

This API uses the following error codes:

- `400 Bad Request:` Resource not found for missing products or invalid values such as negative value of quantity or missing field.
- `401 Unauthorized`: Missing authentication token.
- `500 Internal Server Error`: An unexpected error occurred on the server.

Get All Orders Endpoint

GET /secure/orders/all

Fetches a list of orders from the database.

Response

Returns a JSON object with the following properties:

- `message`: Products fetched successfully.

- `data`: A JSON object of product details
    - `orderId`: The Id of the order.
    - `totalAmount`: The total value of the order.
    - `status`: The status of the order.
    - `products`: The array of JSON objects of products
        - `quantity`: The quantity of products in orderItem.
        - `productId`: The id of the product ordered.
        - `productName`: The name of the product ordered.
        - `rate`: The rate of each product ordered.
        - `totalPrice`: The total value of each product ordered. Value=quantity*rate.

Example

Request:

```
GET /secure/orders/all
```

Body: N/A

Response: `200 OK`

```
{
    "message": "Orders fetched successfully",
    "data": [
        {
            "orderId": 1,
            "totalAmount": 3312005.00,
            "status": "IN_PROGRESS",
            "products": [
                {
                    "quantity": 10,
                    "productId": 1,
                    "productName": "Acer Nitro 2024",
                    "rate": 1200.50,
                    "totalPrice": 12005.00
                },
                {
                    "quantity": 20,
                    "productId": 2,
                    "productName": "Asus Zenbook 2021",
                    "rate": 120000.00,
                    "totalPrice": 2400000.00
                },
                {
                    "quantity": 10,
                    "productId": 3,
                    "productName": "Asus Vivobook 2024",
                    "rate": 90000.00,
                    "totalPrice": 900000.00
                }
            ]
        }
    ]
}
```

Errors

This API uses the following error codes:

- `401 Unauthorized`: Missing authentication token.
- `500 Internal Server Error`: An unexpected error occurred on the server.

CSV Order Import Endpoint

POST /secure/orders/import

Imports ordres from the CSV

Response

Returns a JSON object with the following properties:

- `message`: Success
- `data`: Success or error message.

Request:

```
POST /secure/orders/import
```

Body form-data:

```
key: orderCsv
Value: orderData.csv
```

Response: `200 OK`

```
{
    "message": "Success",
    "data": "Successfully imported 3 orders."
}
```

Errors

This API uses the following error codes:

- `400 Bad Request`: Negative quantity.
- `401 Unauthorized`: Missing authentication token.
- `500 Internal Server Error`: An unexpected error occurred on the server.

CSV Orders Export Endpoint

GET /secure/orders/export

Exports the list of orders available in the database.

Response

Downloads the list of orders as CSV and saves to the user's system.

Example

Request:

```
GET /secure/orders/export
```

Body: N/A

Response: `200 OK`

Errors

This API uses the following error codes:

- `401 Unauthorized`: Missing authentication token.
- `500 Internal Server Error`: An unexpected error occurred on the server.

Delete Order Endpoint

DELETE /secure/orders/delete/{orderId}

Deletes an order by id.

Response

Returns Http Status 204.

Example

Request:

```
DELETE /secure/orders/delete/2
```

Body: N/A

Response: `204 No Content`

Errors

This API uses the following error codes:

- `400 Bad Request`: Order not found for the given ID.
- `401 Unauthorized`: Missing authentication token.
- `500 Internal Server Error`: An unexpected error occurred on the server.